L3: Entry 8 of 10 File: TDBD Mar 1, 1975

TDB-ACC-NO: NN75032867

DISCLOSURE TITLE: Array Logic With Random Access Memory. March 1975.

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, March 1975, US

VOLUME NUMBER: 17
ISSUE NUMBER: 10
PAGE NUMBER: 2867 - 2869

PUBLICATION-DATE: March 1, 1975 (19750301)

CROSS REFERENCE: 0018-8689-17-10-2867

DISCLOSURE TEXT:

3p. Many design problems are encountered by random logic large-scale integration (LSI). Among these are (1) the difficulty in coping with engineering change requirements, (2) the difficulty of interconnecting large numbers of logic circuits in an LSI which allows only a few layers of metallurgy, (3) the severe limitation on the number of logic circuits in a random logic LSI due to the input/output pad requirements, (4) the increased LSI costs where many different part numbers (metallurgy interconnection patterns) are required, and (5) the part number-test pattern proliferation problem. - Array logic or structured logic employing read-only memory (ROM) arrays to perform logic, such as the programmable logic array (PLA) technique, is known to solve some but not all of the above mentioned problems, for instance, part numbers and engineering changes. Array logic employing the conventional read-write or random-access memory RAM approach, however, can address all of the above problems, as herein described. Part number requirements are eliminated since all random-access memory chips are exactly the same, and engineering changes can be achieved by storing different bit patterns into the random-access memory through some auxiliary storage (such as a floppy disk). The only obvious disadvantage of this approach is that the random-access memories are volatile and required bit patterns should be loaded with each machine start. - In Fig. 1. the conventional random-access memory organization of m word by one bit is used to demonstrate the proposed approach. Other organizations with more than one bit output per chip may also be employed. The n inputs which, in conventional random-access memory operation, represent the memory address, are now the logic inputs and control bits. The k outputs represent the result of the logic operation. To accommodate sequential logic requirements, a number of outputs are fed back to the inputs. The feedback loops should be blocked (such as by not applying feedback enable pulse) when the machine starts, during the loading or filling of the bit pattern into the random-access memory array logic. The feedback loops are also blocked in the case of performing combinational logic. - In Fig. 2, an example of a 4-bit up-down counter is shown. The input, a(4), is a control bit (a(4)=1, count-down; a(4)=0, count-up). The remaining inputs, a(3) to a(0), are the input count, with a(0) representing the least significant or 2 bit. The outputs b(3) to b(0), are the output count, with b(0) representing the least significant bit. The 5 inputs and the 4 outputs and the 8 x 4 array in Fig. 2 may be a subset of the n inputs, k outputs, and the m-bit array, respectively, as shown in Fig. 1. Other parts in Fig. 1 may perform other functions. - Fully decoded random-access memory (2/n/ = m) are shown in Figs. 1 and 2. Read write memory in other possible forms, such as extending the present ROM-PLA into random-access memory PLA, may also serve

the same purpose. - The write control, in Figs. 1 and 2, is used only for initial storing when the machine is being turned on. The required bit patterns coming from an auxiliary storage will be fed into the memory data register (Fig. 1) and then written into those k arrays. One of them (random-access memory #j) is shown in greater detail inside the dashed-line box, Fig. 1. The number of feedback loops, p, ranges from zero up to n or k, whichever is smaller. For the sake of clarity, the feed-back loops in Fig. 2 are not shown completely. Also omitted are the inputs, the row, and the column decoders of random-access memories #2, #3 and #4.